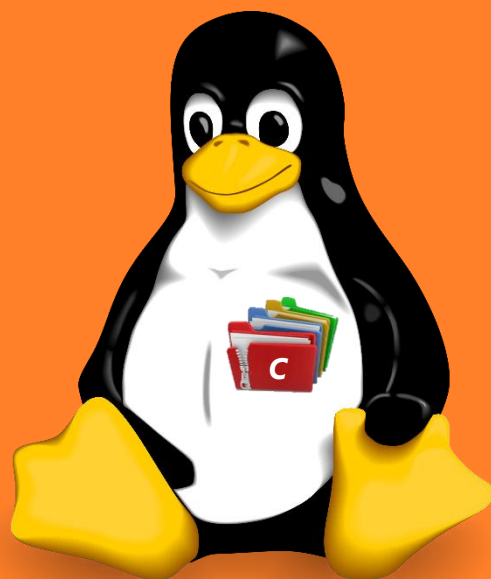




Лабораторная работа #7

# Основы файлового ввода-вывода в Linux. Использование стандартной библиотеки C

*[stdio library, buffered and formatted i/o, exception handling]*



## ЛАБОРАТОРНАЯ РАБОТА #7

# Основы файлового ввода-вывода в Linux. Использование стандартной библиотеки C

### Цель работы

Изучить фундаментальные основы и концепции файлового ввода-вывода в Linux, основные функции ввода-вывода стандартной библиотеки языка программирования C, а также закрепить навыки применения данной библиотеки при написании системных программ.

### Требования

- 1) Разработать многофайловый консольный проект на C/C++ согласно варианту задания с использованием шаблона (паттерна) проектирования MVC.
- 2) Предусмотреть ещё один способ инициализации данных и вывода результата работы программы – с помощью файлового ввода-вывода. Программа должна дополнительно выводить на консоль исходные данные и конечный результат.
- 3) Функции бизнес логики и ввода-вывода должны быть сгруппированы и вынесены в соответствующие библиотеки. Рекомендуется использовать динамические библиотеки.
- 4) ЗАПРЕЩАЕТСЯ в программе использовать под любым предлогом ГЛОБАЛЬНЫЕ переменные!
- 5) Для повышения производительности программы и закрепления навыков работы с памятью везде, где это возможно, необходимо использовать ДИНАМИЧЕСКОЕ выделение и освобождение памяти, а также осуществлять работу через УКАЗАТЕЛИ.
- 6) Каждое задание оформить в виде отдельной бизнес-функции. Все функции должны быть сгруппированы по соответствующим отдельным файлам и вынесены в отдельную библиотеку.

- 7) Все функции должны быть самодостаточные, т.е. при их разработке необходимо придерживаться принципа ***Single Responsibility Principle***.
- 8) При выполнении задания разрешается использовать *IDE*, а также задействовать любой текстовый редактор (к примеру, ***gedit***) и набор компиляторов GNU Compiler Collection (GCC), в частности, компиляторы языков программирования C/C++ ***gcc/g++***, а также утилиту для создания файлов-архивов ***ar***.
- 9) Для автоматизации сборки проекта необходимо использовать утилиту-автосборщик – ***GNU make***.
- 10) При разработке программ придерживайтесь соглашений по написанию кода на C/C++ (Code-Convention).
- 11) Контрольные вопросы по лабораторной работе и ответы на них должны быть записаны в конспект.

## Основное задание

Необходимо расширить функционал любой из программ, созданных в предыдущих лабораторных работах:

- ✓ возможность задавать размеры матрицы и её элементы с помощью указанного файла;
- ✓ возможность выводить результат в файл.


Входной файл может содержать совокупность строк. Каждая строка файла содержит элементы матрицы. Необходимо предусмотреть как ручное создание файла и его заполнение, так и генерацию файла в автоматическом режиме.

## Индивидуальное задание

В программе необходимо определить структуру сущностей согласно варианту задания (разрешается придумать свою структуру-сущность, предварительно согласовав её с преподавателем). Затем необходимо объявить динамическую структуру данных для хранения переменных данного структурного типа. Далее программа должна предоставлять возможность манипулировать данными элементами (к примеру, осуществить инициализацию контейнера, добавление или удаление элементов, отсортировать контейнер элементов по заданному критерию, осуществить необходимые поиск, подсчёт чего-то, сохранить данные контейнера и т.д.). Для сохранения состояния контейнера и его восстановления использовать файловый ввод-вывод.

1. **Student:** id, Фамилия, Имя, Отчество, Дата рождения, Адрес, Телефон, Факультет, Курс, Группа.
2. **Customer:** id, Фамилия, Имя, Отчество, Адрес, Номер кредитной карточки, Номер банковского счета.
3. **Patient:** id, Фамилия, Имя, Отчество, Адрес, Телефон, Номер медицинской карты, Диагноз.
4. **Applicant:** id, Фамилия, Имя, Отчество, Адрес, Телефон, Оценки.
5. **Book:** id, Название, Автор(ы), Издательство, Год издания, Количество страниц, Цена, Переплет.
6. **House:** id, Номер квартиры, Площадь, Этаж, Количество комнат, Улица, Тип здания, Срок эксплуатации.
7. **Phone:** id, Фамилия, Имя, Отчество, Адрес, Номер кредитной карточки, Дебет, Кредит, Время городских и междугородных разговоров.
8. **Car:** id, Марка, Модель, Год выпуска, Цвет, Цена, Регистрационный номер.
9. **Product:** id, Наименование, UPC, Производитель, Цена, Срок хранения, Количество.
10. **Train:** id, Пункт назначения, Номер поезда, Время отправления, Число мест (общих, купе, плацкарт, люкс).
11. **Bus:** Фамилия и инициалы водителя, Номер автобуса, Номер маршрута, Марка, Год начала эксплуатации, Пробег.
12. **Airlines:** id, Пункт назначения, Номер рейса, Тип самолета, Время вылета, Дни недели.
13. **Subject:** id, Название курса, Описание курса, Общее количество часов, Количество лекций, Количество лабораторных работ.
14. **Ticket:** id, Дата сеанса, Время сеанса, Название фильма, Номер ряда, Номер места.
15. ...

*Best of LUCK with it, and remember to HAVE FUN while you're learning :)*

 Victor Ivanchenko



## Контрольные вопросы

1. Устройство и структура жёсткого диска? Что такое главная загрузочная запись (*Master Boot Record, MBR*)? Что такое загрузочный сектор (*Boot Record, BR*)?
2. Последовательность действий от включения компьютера до старта ОС?
3. Что такое файл (*File*)?
4. Что такое файловая система (*File System*) и зачем она нужна?
5. Основные цели файловой системы?
6. Классификация файловых систем?
7. Основная и **главная философия** системного программирования в Linux?
8. Что такое файловый дескриптор (*File Descriptor*) и зачем он нужен?
9. Обычные файлы (*Regular Files*) в Linux и их основные метаданные: позиция в файле (*File Position*), или смещение (*File Offset*), размер файла (*File Length*), имя файла (*Filename*), структура информационный узел (*Information Node, inode*)?
10. Специальные файлы (*Special Files*) в Linux?
11. Что такое каталог (*Directory*)? Зачем нужен корневой каталог (Root Directory)?
12. Что такое ссылка (*Link*)? Какие бывают типы ссылок?
13. Понятие буфера, буферизованный и расширенный файловый ввод-вывод?
14. На какие основные части условно делиться модель файлового ввода-вывода?
15. Абстракция файла в языке программирования C? Стандартные файловые указатели?
16. Функции стандартной библиотеки C (*stdio*) для открытия и закрытия файлов, а также чтения и записи данных: *fopen* (*fdopen*), *fread*, *fwrite*, *fclose* (*fcloseall*), *fflush*, *fseek*, *fgetc* (*getc* or *getchar*), *fputc* (*putc* и *putchar*), *fgets*, *gets*?
17. Форматированный ввод-вывод: *printf* (*fprintf* и *sprintf*), *scanf* (*fscanf* и *sscanf*)?
18. Механизм произвольного доступа к данным с использованием языка C?
19. Функция определения конца файла?
20. Функция стандартной библиотеки языка C для обработки ошибок файлового ввода-вывода?